MARKED UP VERSION OF SPECIFICATION FOR RCE PATENT APPLICATION ENTITLED:

APPLICATION FOR LETTERS PATENT OF THE UNITED STATES OF AMERICA

By:

Steven G. Smarsh
606 Wellington Crescent
Mt. Clemens, MI 48043

and

Brian M. Gehrke
19406 Nicke
Clinton Township, MI 48035

both being citizens of the United States of America

For:

GRINDING MACHINE, COMPUTER SOFTWARE TO OPERATE SUCH A MACHINE, AND THEIR USES THEREFOR

GRINDING MACHINE, COMPUTER SOFTWARE TO OPERATE SUCH A MACHINE, AND THEIR USES THEREFOR

This patent application claims the benefit of prior filed copending U.S. Provisional Patent Application No. 60/090,233 filed on 22 Jun 1998, which Provisional Patent Application is incorporated by reference herein.

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention relates generally to a computer controlled grinding machine for grinding and shaping workpieces and, more particularly, to an extremely accurate compact grinding machine controlled by a computer that writes its own programs.

2. Description of the Related Art

Grinding of workpieces into desired shapes has been desirable for hundreds of years, and as such there are many different types of grinding. Grinding operations as contemplated in the present invention may include, among others, the following operations: grinding, cut-off, lathe, two-axis grinding, OD grinding, ID grinding, turning and centerless grinding.

Centerless grinding is a well-known art useful for grinding rotatable workpieces to high precision. Conventional centerless grinding operations use a workpiece support apparatus and a grinding wheel apparatus, both of which are well-known in the art.

In conventional apparatuses, the grinding wheel apparatus typically includes a wheel spindle head which is vertically movable, i.e., movable up and down along the "y" axis, a means such as a ball or lead screw which is driven by an y-axis motor for moving the wheel spindle head, a spindle extending through and beyond the wheel spindle head, a grinding wheel mounted on the spindle, and a motor for rotating the spindle and grinding wheel.

Prior art patents USPN 5,746,643 and 5,746,644 disclose numerically controlled grinding machines. Although these patents disclose useful grinding machines, they must be individually programmed for each cutting or grinding operation.

One exceptional workpiece support apparatus is disclosed in U.S. Patent No. 5,121,571, issued June 16, 1992, to Smarsh, which patent is incorporated herein by reference. The workpiece support apparatus of the Smarsh patent includes (among other beneficial components) a table, a base supported on the table, a carriage supported by the base, a horizontally-oriented spindle extending through and beyond the carriage, a regulating roller mounted on the spindle, and a means for rotating the spindle which thereby rotates the regulating roller. The regulating roller provides a rotating surface for supporting and rotating the workpiece.

Typically, the conventional regulating rollers are formed from a porous material so that micro-fine dust on the roller will not harm the friction characteristics and the workpiece may be rotated at a consistent speed. However, when the dust builds up to a point where the friction surface is too smooth to perform rolling operations, the surface must be roughened up to allow it to grip the workpiece during the grinding operation.

The table of the workpiece support apparatus is movable along a horizontal axis, or side-to-side, which will be referred to as the "z" axis. The table may be moved along the z axis by a z-axis motor which turns a belt/pulley/ball/or lead screw mechanism as is known in the art. The z-axis motor may be controlled by a conventional motion controller, such as Parker's COMPUMOTOR OEM 6200 motion controller, available from Parker Corporation of Rohnert Park, California.

By controlling the y-axis movement and the z-axis movement in the grinding apparatus, a workpiece may be ground as desired to meet various conformations. Current methods of controlling the y- and z-axis movement include the use of computer systems with individually written computer programs to direct the motion of the grinding apparatus. However, in these current methods, a highly-trained computer programmer writes a program template in programming language and a corresponding template in machine language so that the machine will perform based on the content of the program template. The grinding apparatus is then operated on request by running the computer program.

The accuracies with which these grinding operations may be performed is desirably as great as possible. Although the abovementioned Smarsh patent reveals a grinder capable of reproducibly grinding to several millionths of an inch, the present invention goes beyond that machine's capabilities in that the present invention can reproducibly, and nearly instantaneously achieve that result consistently. Furthermore, with the new computer programming capabilities, varying grinding operations, including dressing operations, can be performed on the same machine with minimal time being expended for the change of the routine.

It should be noted that forms can be ground into the regulating roller for various shaped parts by using the computer program and the grinding wheel of the present invention. In addition, the computer program can be used to shape or form conventional wheels with a single point diamond tool. This capability of the present machine to grind with such precision may be put to advantage in many ways, and should not be limited by the discussion within.

In conventional machines, if a variation of the computer program is desired, e.g., if it is desired to have the grinding wheel position slightly different, a completely new program needs to be written by the trained computer programmer. This is a very time consuming procedure, and must be done for each and every operation desired. Downtime is experienced while waiting for the computer programmer to intervene before the new cutting operation can be commenced.

The need to prepare individual computer programs for each operation does not lend itself well to instantaneous running by an operator. In direct contrast, the present invention discloses a computer-machine combination which writes its own programs for desired operations, and then is capable of using that new program to control the grinding machine. This saves time, money and aggravation.

Another problem which is addressed by the present invention entails a new apparatus and a new method of programming a dressing operation. One can note that after several operations of grinding, the surfaces of the grinding wheel have rough areas or uneven areas which will result in poor quality grinding. To maintain the grinding wheel surface, the grinding wheel needs to be evened out or "dressed".

In the past, in order to dress the grinding wheel, the grinding wheel was manually positioned above a rotating dressing wheel to grind the surface of the grinding wheel. Then, the

grinding wheel was repositioned above the workpiece support apparatus. When the grinding wheel was repositioned after the dressing operation, it was extremely difficult to return the grinding wheel to the exact same position relative to the workpiece support apparatus, so some accuracy is lost when this act is performed. Maintaining millionths of an inch accuracy in this scenario became nearly impossible for even the most experienced dressing operator.

In view of the above-described problems with current methods of the prior art grinding and dressing techniques, the present invention seeks to provide an easy-to-use computer system for nearly instantly constructing its own individual computer programs capable of operating various grinding apparatuses.

Another provision being sought by the present invention is an easy-to-use computer system for constructing a computer program capable of operating various grinding apparatuses, which computer system allows for easy alterations to the constructed computer program.

Yet another provision being sought is a computer system for easily, accurately, and quickly operating the dressing of a grinding wheel without removing the grinding wheel from the grinding apparatus.

SUMMARY OF THE INVENTION

In accordance with the present invention, a serious improvement is disclosed for the grinding art to aid grinding operators in achieving extremely tight tolerances consistently, with the ability to shift between various grinding shapes and sizes almost instantaneously. Disclosed is a computer controlled grinding and dressing machine which can write (nearly instantaneously by the operator) its own computer programs with the input of specific data to allow for nearly seamless changes in operation, in addition to being capable of dressing the grinding wheel and regulating roller on site to keep the grinding operation within tolerances on the order of millionths of an inch.

Dressing of diamond and borazon wheels are made possible with the present machine in that all types wheel can be dressed. One of ordinary skill in the art can appreciate that grit wheels can be form dress as well on this machine. Any form can be made with the computer program.

Furthermore, there is disclosed a new embodiment of the dressing wheel, utilizing a "sawblade" sandwiched between two dressing stone portions. This new "sandwich" of a twin dressing stone/sawblade combination is especially useful for dressing and surface treating diamond or borazon wheels, as they can tend to "glaze" over with traditional dressing stones by themselves.

Yet one more aspect of the present invention is the full implementation of Windows-based software, including a Window for value increment increase or decrease while setting parameters. This alleviates the need for dial turning or other manual input of data as in the prior art.

Other advantages of the present invention will be readily appreciated as the same becomes better understood after reading the subsequent description taken in conjunction with the appendant drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1A is a representation of a computer controlled grinding machine according to the present invention;

FIG. 1B is a schematic of the computer input as it relates to the grinding machine and its components;

FIG. 1C is a side elevation view of the grinding machine in operation.

FIG. 2 is a start-up screen for the program menu operation;

FIG. 3 is a program name screen;

FIG. 4 is a create program screen;

FIG. 5 is a move icon screen;

FIG. 6 is an edit icon values screen for Axis 1;

FIG. 7 is an edit icon values screen for Axis 2;

FIG. 8 is a shut down screen for the operation;

FIG. 9 is a dress wheel routine screen;

FIG. 10 is a reminder screen for the dressing;

FIG. 11 is a regulating roller dress routine screen;

FIG. 12 is a run program menu screen for increasing or decreasing the tolerances of position;

FIG. 13 is a goto screen for the home machine;

FIG. 14 is an edit abort screen for setting the velocity of the abort;

FIG. 15 is a programming menu screen to show how to change between operations;

FIG. 16 is a value set screen;

FIG. 17 is a radius and shape of an example part;

FIG. 18 is a side elevational view of a sawblade in accordance with the present invention; and

FIG. 19 is a side elevational view of a dressing stone/sawblade sandwich configuration in accordance with the present invention.

DETAILED DESCRIPTION OF THE INVENTION

In accordance with the present invention, there is disclosed a computer controlled grinding machine capable of reproducibly grinding workpieces with extremely high tolerances on the order of millionths of an inch. The computer controlled grinding machine is also capable of changing from one grinding operation to another desired grinding operation without the interference of a computer programmer as this computer program can write its own grinding computer programs with the input of certain key data.

This entire operation can be done within the compact grinding machine as disclosed and claimed herein. As disclosed in the earlier patent USPN 5,121,571 of one of the same inventors, a compact grinding machine can be used for centerless grinding. The additions of this invention include the computer control, the dressing wheel, and the incremental value box. Not only is the present invention more advantageous over the prior art because of its computer control system, there is also included a novel dressing wheel for keeping the tolerances in check, as well as an integrated value increment change box to further aid in accuracy. Furthermore, the present invention is a nice, neat compact unit that can fit into almost any shop for working.

FIG 1A illustrates the connection between a computer 100 located within an electrical console 102 and a grinding machine 10. As must be noted, the grinding machine of the present example and figures can be interchanged with any other rotational operation, such as cut-off operations, lathe work, OD grinding, ID grinding, turning, besides centerless grinding. As one will be able to appreciate upon further review, the computer program disclosed herein may be used to advantage with any of these rotational operations without undue experimentation on the part of one of ordinary skill in the art. However, for ease of explanation, this discussion will be confined to centerless grinding.

Grinding machine 10 is located on rollers 12 and 13. The other parts are described and called out in incorporated patent number 5,121,571. A dressing wheel 104 is now included, placed on the extending portion of the spindle. The grinding wheel (shown in FIG 1C) is a separate wheel that is co-planar with the dressing wheel and the regulating roller. It can be lowered down onto the workpiece as the regulating roller 24 rotates the part to be ground by the grinding wheel.

Grinding machine 10 can hold the workpiece to be ground entirely on the regulating roller to effect centerless grinding, or the end of the workpiece can extend over the end for plunge grinding, form grinding (if the grinding wheel has been formed), and infeed grinding.

Due to the design of the grinding machine, no indicating is necessary. Parts being ground can be removed from the regulating roller, inspected, and then returned to the regulating roller by hand without any indicating, while still having part repeats within millionths.

In order to effect the dressing operation of the grinding wheel, the grinding wheel is jogged forward of the regulating roller to contact the dressing wheel attachment, and the computer can regulate how the grinding wheel is dressed. Dressing of the regulating roller is made easy by pivoting back the work blade 38.

FIG 1B is a schematic diagram showing the interrelationship between the data input/monitor/IO interface board/data processor and the motion controller. Also, for purposes of discussion here, we will only be discussing the use of a keyboard, although it is thoroughly contemplated that a mouse may be used with equal efficacy. Consequently, all mention of data input or keyboards will be immediately interpretable to cover all forms of data input, whether it be keyboard, mouse, voice command, or the like.

FIG 1C is a side view of the grinding machine of the present invention with workpiece 80 shown in place between the grinding wheel 14, the regulating roller 24, and the work blade 38.

FIG. 2 is a schematic representation of the interrelationships between the data processor and the grinding machine in skeleton. Y axis motor 110 is in communication with spindle head 112, which is in turn, attached to the spindle 114. Grinding wheel 116 is attached to the spindle 114, and may be lowered down onto a workpiece (not shown). Regulating roller 118 and carriage 124 are attached to spindle 122, as is the dressing wheel 120. Base 126 can be moved along the z-axis by z-axis motor 130 in order to effect the desired grinding.

With combined reference now to FIG's 3-17, an example will be detailed of the computer programming control of the grinding machine, while all the steps are taking place. Note that all, if not nearly all, of the data input can be done with a mouse. This is a huge advantage over

the prior art systems which required the use of the keyboard. As many machinists are uncomfortable with the use of keyboards, the nearly exclusive use of a mouse is key to marketability.

FIG. 3 indicates a smiley face and a test program. At the bottom of the screen in a "shut down" command to shut down the machine and the program. The smiley face is the last program ran.

In order to write a new program, the Program Menu screen will display on the monitor as shown in FIG. 3. Clicking on the word Program on the top tool bar displays the pull-down menu, and clicking on New allows you to select the icon that most closely resembles the profile of the part you would like to create. By making such a selection, and entering data as can be seen hereinbelow, the computer will assign certain characters to certain input, and numerical algorithms and/or computer programs will automatically be programmed into the computer.

FIG. 4 shows the window with the "Write program" box, and clicking on that window will begin creation of the program. Note the MOVE ICONS at the top of the screen. The operator then selects the "MOVE ICONS" needed for running the desired grinding operation. FIG. 5 shows that once a MOVE ICON has been chosen, it is possible to change the order of the process. The properties of the MOVE display on the right side of the programming screen. FIG. 6 shows the screen used to edit the values of an icon. The values may be varied by either scrolling to the number desired, or by typing in the value on a keyboard. It is advisable to "home" the machine before proceeding further. This will re-zero the machine and gives the operator time to place the workpiece in the unit with the grinding wheel off~~.~~ ~~B~~by jogging the grinding wheel down to the top of the workpiece. While manually spinning the wheel, contacting the top of the part ensures a close fit. Thereafter, the wheel should be raised up about two thousandths of an inch. This sets Axis 1 in the program.

FIG. 7 shows a screen for setting the Axis 2 values. This is done by jogging the base of the machine out until the workpiece has cleared the wheel. By setting the grinding wheel down just in front of the workpiece, Axis 2 can be set. By pressing the window for Send and Run, the machine is set for grinding. By turning on the grinding switch, the coolant switch and cycle button on the control panel, the grinding operation begins~~.~~ ~~I~~if the result is totally desirable,

unloading the newly ground part and replacing it with a new workpiece gets it ready to start another cycle.

FIG. 8 is the screen for shutting down the operation. By clicking on the STOP button on the previous screen, a new SHUT DOWN pop-up menu appears. By selecting "Shut Down Machine", the machine can be turned off. This constitutes proper shutdown, and should not pose any danger to the computer, the information on the hard drive, nor should any computer lock-up be experienced.

FIG. 9 illustrates the dressing operation and the computer screen that goes along with it. The Run Program screen has a button to locate the Dress Routine Tab. Clicking on it causes a window to pop up. Select either the Wheel Dress Routine or the Regulating Roller Dress. Making sure to check the direction of the reverse sweep allows the operator to set it up to either plunge down and sweep back or to rise up and sweep back. By selecting Send and Run, the Dress Routine dialog box (Fig. 10) pops up, which reminds the operator to do certain safety precautions.

The dressing operation can begin on-site as the grinding wheel is jogged back onto the dressing wheel. The operator merely lowers the wheel until it slightly touches the dressing wheel, and then he moves the wheel toward the unit until it is clear from the dressing wheel. This sets the parameters for the cycle, and the wheel will now sweep across the dressing roller at predetermined amounts. The initial sweep has an amount of 01.50000" at a forward velocity of 05.00000", raising up 00.00200", and moving back 01.50000" at a reverse velocity of 10.00000". Then it lowers down 00.00300" and sweeps across the dressing roller.

FIG. 11 shows the routine for dressing the regulating roller. Again, going to the Run Program Screen will locate the Regulating Roller Dress Routine Tab. As before, clicking on it will cause a window to pop up, and similar activities as before will take place.

FIG. 12 is the screen which is used for increasing or decreasing the tolerances of position. By putting the number in the value box and clicking on the + or the - sign, the tolerance will either increase or decrease. After selecting the positions that this will apply to, the changes take place via a computer program that the computer writes itself. The computer writes these programs by correlating certain characters that it assigns to various values or configurations to

further preselected general shapes. However, by utilizing the characters correlated to various values input, individual computer programs are written.

FIG. 13 is the GoTo screen which is used after a grinding job has already been set up and the machine is at the original home position. By using the set-up values to position Axis 1 and Axis 2, the job can be recreated. The Abort button can be used at any time during any program. To use the abort during a program, hitting the space bar on the keyboard will abort the cycle. FIG. 14 is the screen to be used to edit the abort permitting the operator to set the velocity and the position that is desired for use during the abort.

FIG's 15-17 indicate the screens for an illustrative example of a precision grinding to be performed on a half inch diameter by three inch long piece of graphite. After the Program menu appears, create and name a new program. For this example, we will name the program "RADIUS". Clicking on the 17th icon will help to visualize the part. Clicking on the Create Program button, and the Programming window will pop up. By entering the MOVE ICONS as shown in FIG. 15, the radius will be at 50% to 100% (which is the default setting). Enter the values as shown in FIG. 16, and click on the "Write" the program and test it. FIG. 17 shows the resultant part that has a dip, then a taper with a diameter of 0.4630 and the radius that ends at the tip.

Another aspect of the present invention includes a new dressing stone configuration which has increased efficacy, which is especially useful for diamond and borazon wheels. FIG. 18 shows the new "sawblade" design of dressing wheel component 200 with sawblades 202 and a central orifice 204 for attachment to a spindle. FIG. 19 shows a side elevational view of the dressing wheel with a "sandwich" of dressing stones 206 over and under the sawblade 202, again surrounding an orifice 204 for attachment. The teeth of the sawblade act to rip off and roughen the surface of a wheel being dressed without glazing or heat expanding the wheel being dressed.

Therefore, there is a new computer controlled grinding machine in accordance with the present invention that will write its own computer programs, provide on-site dressing, and perform its tasks by utilizing a mouse. Furthermore, the value increments can be scrolled up and down without having to type in values. The appendant claims will further define the invention.

INDUSTRIAL APPLICABILITY OF THE INVENTION

The industrial applicability of the present invention includes use for grinding workpieces to be used in various industries, and also includes use for the preparation of extremely close tolerance machined workpieces.

CLAIMS

What is claimed is:

1. (withdrawn) ~~A workpiece support apparatus for centerless grinding, comprising:~~

~~a carriage having an edge;~~

~~a rotatable spindle extending through the carriage and having an extending portion, the extending portion extending beyond the edge of the carriage;~~

~~a regulating roller mounted on the extending portion of the spindle, the regulating roller for supporting and rotating the workpiece;~~

~~a wheel dressing roller having a polishing outer surface and being mounted on the extending portion of the spindle distal of the carriage relative to the regulating roller; and~~

~~means for rotating the spindle which thereby rotates the regulating roller and the wheel dressing roller.~~

2. (withdrawn) ~~The support apparatus according to claim 1, wherein the wheel dressing roller has an outer diameter which is less than the outer diameter of the regulating roller.~~

3. (currently amended) A computer program in combination with a computer and a rotational grinding apparatus having a grinding wheel, a blade and a regulating roller for enabling a user through a user interface to control the rotational grinding apparatus to dress at least one of the grinding wheel, the blade, and the regulating roller by using a wheel dressing roller supported on a spindle and grind a workpiece, comprising:

means for displaying a computer screen on a monitor and for selecting an icon resembling a desired profile for dressing, said screen having at least one numerical data input value display window on the screen, wherein the value displayed in the at least one numerical data input value display window corresponds to a numerically controlled pattern of dressing, and wherein said pattern of dressing is variable by changing the value in the display window, whereby

the computer will assign certain characters to certain input, and numerical algorithms and computer programs will automatically be programmed into the computer;

means for displaying a computer screen on a monitor and for selecting an icon resembling a desired profile for grinding, said screen having at least one numerical data input value display window on the screen, wherein the value displayed in the at least one numerical data input value display window corresponds to a numerically controlled pattern of grinding, wherein the value displayed in the at least one numerical data input value display window corresponds to a numerically controlled pattern of grinding, whereby the computer will assign certain characters to certain input, and numerical algorithms and computer programs will automatically be programmed into the computer, and wherein said pattern of grinding is variable by changing the value in the display window;

means for accepting a value from the user and displaying the value in the value display window;

means for setting at least two grinding axes; and

means for initiating a grinding operation, coolant, and cycles;

whereby selecting a value in the at least one value display window automatically directs the computer program to select a scripted computer program to control the grinding apparatus to in-situ dress and grind at least one of the grinding wheel, the blade and the regulating roller, and a workpiece in a desired configuration.

4. (previously presented) The program according to claim 3, further comprising a means for changing the value in the at least one numerical data input value display window to increase or decrease the values displayed by using a mouse to scroll up and down a value list.

5. canceled

6. (currently amended) A computer program in combination with a computer and a rotational grinding apparatus having a grinding wheel, a blade and a regulating roller for

enabling a user through a user interface to control the rotational grinding apparatus to dress the regulating roller and to grind a workpiece, comprising:

means for displaying a computer screen having at least one numerical data input value display window, wherein the value displayed in the value display window corresponds to a numerically controlled pattern of regulator dressing whereby the computer will assign certain characters to certain input, and numerical algorithms and computer programs will automatically be programmed into the computer, and wherein said pattern of dressing is variable by changing the value in the display window relating to a variable in the process of dressing the regulating roller;

means for displaying a computer screen having at least one numerical data input value display window, wherein the value displayed in the value display window corresponds to a numerically controlled pattern of grinding, whereby the computer will assign certain characters to certain input, and numerical algorithms and computer programs will automatically be programmed into the computer, and wherein said pattern of dressing is variable by changing the value in the display window relating to a variable in the process of grinding;

whereby selecting numerical data input values in the value display windows automatically directs the computer to select a scripted computer program to control the grinding apparatus to in-situ dress and grind the regulating roller and the workpiece in a desired configuration.

7. (previously presented) The program according to claim 6, further comprising means for changing the value in the value display window to either increase or decrease the values.

8. (previously presented) The program according to claim 6, further comprising means for enabling modification of the accepted value.

9. (withdrawn) ~~A grinding apparatus capable of dressing a grinding wheel using a computer system having a user interface, the apparatus comprising:~~

~~a user interface including a means for data input and a monitor;~~

~~an input/output (I/O) interface board electronically connected to the user interface;~~

~~a data processor electronically connected to the I/O board;~~

~~a motion controller electronically connected to the I/O board;~~

~~a y-axis drive unit electronically connected to the motion controller;~~

~~a z-axis drive unit electronically connected to the motion controller;~~

~~a y-axis motor electronically connected to the y-axis drive unit;~~

~~a z-axis motor electronically connected to the z-axis drive unit;~~

~~a grinding wheel;~~

~~a rotatable wheel dressing roller;~~

~~the combination of the grinding wheel and the wheel dressing roller being connected to the y-axis motor and the z-axis motor such that the y- and z-axis motors can cause the grinding wheel and the wheel dressing roller to contact each other so that when the grinding wheel and the wheel dressing roller are rotating, the wheel dressing roller will smooth the surface of the grinding wheel;~~

~~the data processor for controlling the y-axis motor to control the depth of grinding and for controlling the x-axis to control the grinding wheel and the wheel dressing roller to sweep past one another longitudinally.~~

10. (withdrawn) ~~A centerless grinding apparatus capable of dressing a regulating roller using a computer system having a user interface, the apparatus comprising:~~

~~a user interface including a means for data input and a monitor;~~

~~an input/output (I/O) interface board electronically connected to the user interface;~~

~~a data processor electronically connected to the I/O board;~~

~~a motion controller electronically connected to the I/O board;~~

~~a y-axis drive unit electronically connected to the motion controller;~~

~~a z-axis drive unit electronically connected to the motion controller;~~

~~a y-axis motor electronically connected to the y-axis drive unit;~~

~~a z-axis motor electronically connected to the z-axis drive unit;~~

~~a grinding wheel;~~

~~a regulating roller;~~

~~the combination of the grinding wheel and the regulating roller being connected to the y-axis motor and the z-axis motor such that the y- and z-axis motors can cause the grinding wheel and the regulating roller to contact each other so that when the grinding wheel and the regulating roller are rotating, the grinder wheel will smooth the surface of the regulating roller;~~

~~the data processor being designed for controlling the y-axis motor to control the depth of grinding and for controlling the x-axis to control the grinding wheel and the regulating roller to sweep past one another longitudinally.~~

11. (currently amended) A computer system with a computer screen for enabling the selection of a computer program by a user utilizing value display windows on the computer screen, the computer program being adapted and created for controlling a rotational grinding apparatus, the computer system also for compiling data so that the rotational grinding apparatus performs as the user specifies, the computer system comprising:

means for displaying a computer program screen on a monitor selecting an icon resembling a desired profile for dressing and grinding, whereby the computer will assign certain characters to certain input, and numerical algorithms and computer programs will automatically be programmed into the computer, said screen having at least one numerical data input value display window on the computer screen which relates to a numerical variable in the process of grinding with a rotational grinding apparatus;

means for accepting a numerical value from the user and displaying the numerical value in the value display window;

means for setting at least two grinding axes; and

means for initiating a grinding operation, coolant, and cycles;

means for sending the accepted value to a computer program for operating the grinding apparatus; and

means for directing the computer to assign certain characters to certain input, and numerical algorithms and computer programs will automatically be programmed into the select a script computer program from the previous computer program containing the accepted value such that the computer program controls the grinding apparatus in a desired manner reflecting the numerical values entered into the value display windows.

ABSTRACT

An extremely precise computer controlled grinding machine having the capability to write its own computer programs for controlling the grinding machine to perform individual tasks, including both grinding and dressing to several millionths of an inch. A new device for dressing is included which is directly on the machine, alleviating the need for a separate dressing machine. The Windows-based computer program is easy to use, and includes numerous features for self-maintenance and reproducibility.

CHAPTER ONE

Programming Fundamentals

IN THIS CHAPTER

This chapter is a guide to general 6K programming tasks. It is divided into these main topics:

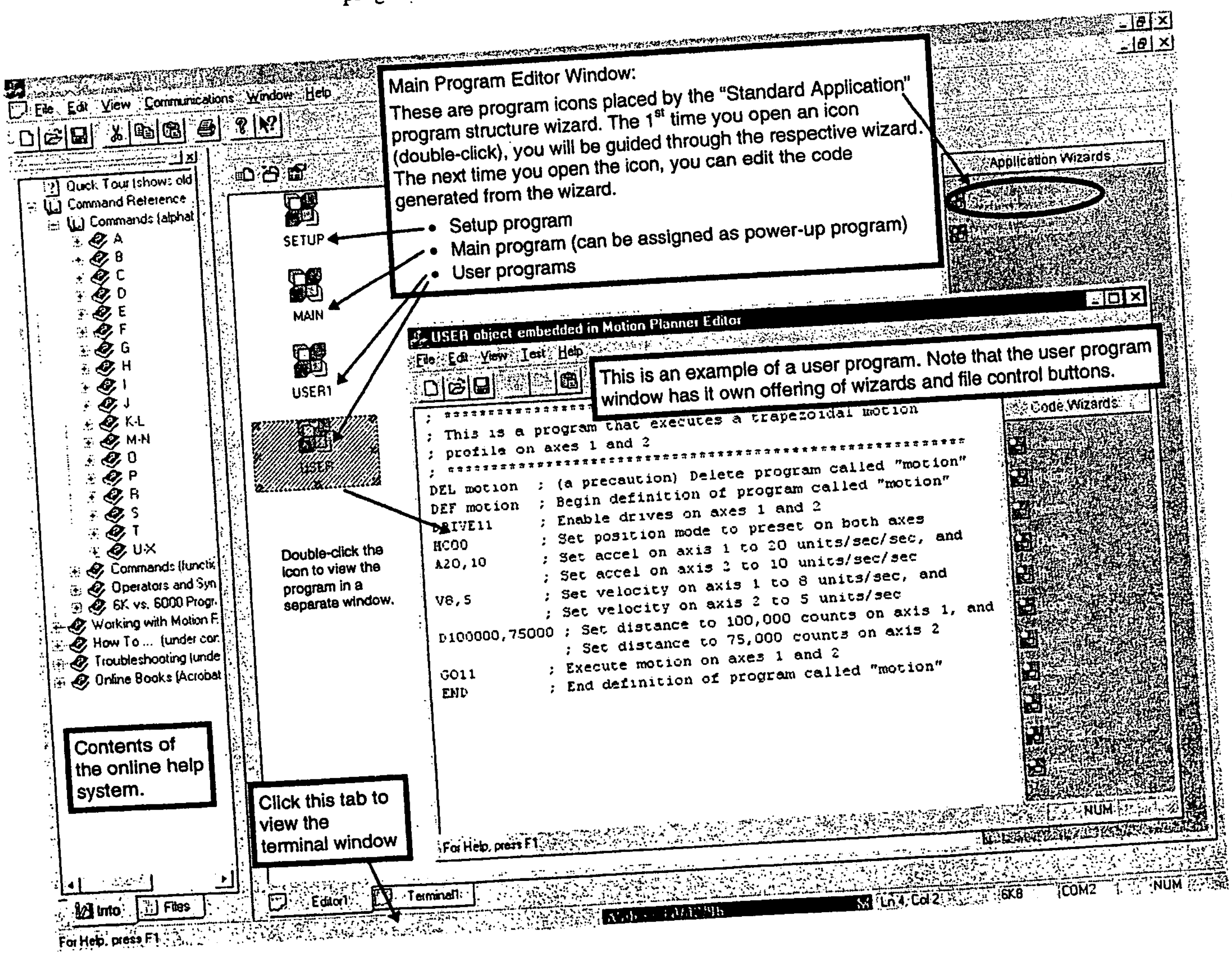
Motion Planner Programming Environment

Every 6K Series controller is shipped with Motion Planner, a Windows-based programming tool designed to simplify your programming efforts. The Motion Planner interface allows you to:

- Create, edit, download, and upload programs (or code modules).
- Tune your system to optimize performance.
- Test & debug programs and controller operation with customizable displays.
- Organize all of your programs and resource files for your programming project.

PROGRAMMING SUPPORT. To help you program with speed and efficiency, Motion Planner provides these "performance support" features:

- **Smart Editor**: The smart editor is the focal point for your programming tasks: The smart editor watches over your shoulder and provides syntax checking on the fly (as you type). To get detailed information on the command you're using, just press the F1 key. At any point, you can check the entire program file for logic flow and syntax errors.
- **Programming Help with Wizards**: While you are working in the Editor, you can use the wizards to speed up your programming tasks and minimize your need to learn the details of the programming language. Wizards are available for general program structure, general system setup (including servo tuning), error programming, and a host of other programming tasks.



Command Syntax

Introduction

The 6K programming language accommodates a wide range of needs by providing basic motion control building blocks, as well as sophisticated motion and program flow constructs.

The language comprises simple ASCII mnemonic commands, with each command separated by a command delimiter (carriage return, colon, or line feed). The command delimiter signals the 6K product that a command is ready for processing.

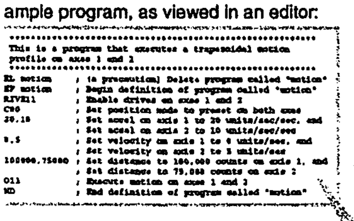
Upon receiving a command followed by a command delimiter, the 6K controller places the command in its internal command buffer, or queue. Here the command is executed in the order in which it is received. To make the command execute immediately, place an exclamation point (`!`) in front of it (e.g., The `TAS` command will be executed after all commands ahead of it in the command buffer are executed; but `!TAS` will execute before any other commands in the command buffer).

Spaces and tabs within a command are processed as neutral characters. Comments can be specified with the semicolon (`;`) character — all characters following the semicolon and before the command delimiter are considered program comments.

Some commands contain one or more data fields in which you enter numeric or binary values or text:

- Numeric data fields. For example, `A20,10` is an acceleration (`A`) command that sets the acceleration for axes 1 and 2 to 20 units/sec^2 and 10 units/sec^2, respectively.
- Binary fields. For example, `DRIVE1011` is a drive enable (`DRIVE`) command that enables axes 1, 3 and 4 and disables axis 2.
- Text fields. For example, `STARTPpowrup` is a startup program assignment (`STARTP`) command that assigns the program called "powrup" as the startup program to be executed automatically when the 6K product is power up or reset.
- To check what the data field settings are for a particular command, simply type in the command without the data fields. The 6K will display the command settings. For example, after executing the `A20,10` noted above, you could type in the `A` command by itself and the 6K controller would respond with `A20,10`.

ample program, as viewed in an editor:



These are command line comments, comprising a semi-colon and text. The comments are separated from the command by a tab. A carriage return is placed at the end of each command line.

`DEL motion` — Text field; Command name

`DRIVE 11` — Binary data field (corresponds to axes 1 & 2, from left to right); Command name

```
; ***********************************************************
; This is a program that executes a trapezoidal motion
; profile on axes 1 and 2
; ***********************************************************
DEL motion        ; (a precaution) Delete program called "motion"
DEF motion        ; Begin definition of program called "motion"
DRIVE11           ; Enable drives on axes 1 and 2
MC00              ; Set position mode to preset on both axes
A20,10            ; Set accel on axis 1 to 20 units/sec/sec, and
                  ; Set accel on axis 2 to 10 units/sec/sec
V8,5              ; Set velocity on axis 1 to 8 units/sec, and
                  ; Set velocity on axis 2 to 5 units/sec
D100000,75000     ; Set distance to 100,000 counts on axis 1, and
                  ; Set distance to 75,000 counts on axis 2
GO11              ; Execute motion on axes 1 and 2
END               ; End definition of program called "motion"
```

specific status bits (e.g., to check axis 2's axis status bit #25 to see if a target zone timeout occurred, type the `2TAS.25` command and observe the response).

Binary and Hex Values

When making assignments with or comparisons against binary or hexadecimal values, you must precede the binary value with the letter "b" or "B", and the hex value with "h" or "H". Examples: `IF(IN=b1xØ1)` and `IF(IN=h7F)`. In the binary syntax, an "x" simply means the status of that bit is ignored. Refer also to *Using Binary Variables* (page 21).

Related Operator Symbols

Command arguments include special operator symbols (e.g., +, /, &, ', >=, etc.) to perform bitwise, mathematical, relational, logical, and other special functions. These operators are described in detail, along with programming examples, at the beginning of the *Command Descriptions* section of the *6K Series Command Reference*.

Programmable Inputs and Outputs Bit Patterns

I/O pin outs, specifications, and circuit drawings are provided in each 6K Series Hardware Installation Guide.

The 6K product has programmable inputs and outputs. The total number of onboard inputs and outputs (trigger inputs, limit inputs, digital outputs) depends on the product. The total number of expansion inputs and outputs (analog inputs, digital inputs and digital outputs) depends on your configuration of expansion I/O bricks connected to the "EXPANSION I/O" connector.

These programmable I/O are represented by binary bit patterns, and it is the bit pattern that you reference when programming and checking the status of specific inputs and outputs. The bit pattern is referenced in commands like `WAIT(IN.4=b1)`, which means wait until onboard programmable input #4 (TRG-2B) becomes active. To ascertain your product's I/O offering and bit patterns, refer to Chapter 3 (page 77).

Creating Programs

Debugging Programs: Refer to page 219 for methods to isolate and resolve programming problems.

A *program* is a series of commands. These commands are executed in the order in which they are programmed. Immediate commands (commands that begin with an exclamation point [!]) cannot be stored in a program. Only buffered commands may be used in a program.

A *subroutine* is defined the same as a program, but it is executed with an unconditional branch command, such as GOSUB, GOTO, or JUMP, from another program (see page 22 for details about unconditional branching). Subroutines can be nested up to 16 levels deep. NOTE: The 6K family does not support recursive calling of subroutines.

Compiled profiles & PLC programs are defined like programs, using the DEF and END commands, but are compiled with the PCOMP command and executed with the PRUN command (PLC programs are usually launched in PLC Scan Mode with the SCANP command). Compiled profiles and PLC programs also affect a different part of the product's memory, called *compiled memory*. A compiled *profile* could be a multi-axis contour (a series of arcs and lines), an individual axis profile (a series of GOBUF commands), or a compound profile (combination of multi-axis contours and individual axis profiles). A compiled *PLC program* is a pre-compiled program that mimics PLC functionality by scanning through the I/O faster than in normal program execution. For information on contouring, refer to page 124; for information on compiled individual axis profiles, refer to page 134; and for information on PLC programs, refer to page 104.

Program Example

The illustration below identifies the elements that comprise the general structure of a program.

Use DEF to begin defining the program.

Use DEL to delete the program (a precaution).

Contents of the program.

Use END to finish defining the program.

```
; *************************************************************
; This is a program that executes a trapezoidal motion
; profile on axes 1 and 2
; *************************************************************
DEL motion      ; (a precaution) Delete program called "motion"
DEF motion      ; Begin definition of program called "motion"
DRIVE11         ; Enable drives on axes 1 and 2
MC00            ; Set position mode to preset on both axes
A20,10          ; Set accel on axis 1 to 20 units/sec/sec, and
                ; Set accel on axis 2 to 10 units/sec/sec
V8,5            ; Set velocity on axis 1 to 8 units/sec, and
                ; Set velocity on axis 2 to 5 units/sec
D100000,75000   ; Set distance to 100,000 counts on axis 1, and
                ; Set distance to 75,000 counts on axis 2
GO11            ; Execute motion on axes 1 and 2
END             ; End definition of program called "motion"
```

These are command line comments, comprising a semi-colon and text.
The comments are separated from the command by a tab.
A carriage return is placed at the end of each command line.

Use the Wizards in Motion Planner

Motion Planner provides wizards that make it easy to create your program. Below is a partial list of the wizards available.

- Application Wizards (for program structure guidance)
  - Standard Application
  - Multitasking Application
- Program Wizards
  - Setup Program
  - Main Program
  - User Program
  - Error Program
- Setup Wizards
  - Drive Setup
  - Feedback Setup
  - Scaling Setup
  - Limit Setup
  - Servo Tuner
  - On-board I/O Setup
  - Expansion I/O Setup
  - Jogging Setup
  - Joystick Setup
  - Variable setup
- Action Wizards
  - Motion
  - Home
  - Output
  - If
  - Loop
  - Wait
  - Assignment
  - Target Zone
  - Registration

Storing Programs

After a program or compiled program/profile is defined (DEF) or downloaded to the 6K controller, it is automatically stored in non-volatile memory (battery-backed RAM). Information on controlling memory allocation is provided below (see *Memory Allocation*).

Memory Allocation

Your controller's memory has two partitions: one for storing *programs* and one for storing *compiled profiles & PLC programs*. The allocation of memory to these two areas is controlled with the MEMORY command.

> **"Programs" vs. "Compiled Profiles & PLC Programs"**
>
> Programs are defined with the DEF and END commands, as demonstrated in the *Program Example* on page 9.
>
> Compiled Profiles & PLC Programs are defined like programs, using the DEF and END commands, but are compiled with the PCOMP command and executed with the PRUN command (PLC programs are usually executed in PLC Scan Mode with the SCANP). A compiled profile could be a multi-axis *contour* (a series of arcs and lines), an *individual axis profile* (a series of GOBUF commands), or a *compound profile* (combination of multi-axis contours and individual axis profiles). A PLC program is a pre-compiled program that mimics PLC functionality by scanning through the I/O faster than in normal program execution.
>
> Programs intended to be compiled are stored in program memory. After they are compiled with the PCOMP command, they remain in program memory and the *segments* (see diagram below) from the compiled program are stored in compiled memory. The TDIR report indicates which programs are compiled as compiled profiles ("COMPILED AS A PATH") and which programs are compiled as PLC programs ("COMPILED AS A PLC PROGRAM").
>
> For information on contouring, refer to page 124; for information on compiled individual axis profiles, refer to *Compiled Motion Profiling* on page 134; and for information on PLC programs, refer to page 104.

MEMORY command syntax (example)

MEMORY80000,70000

80000 — Memory allocation for Programs (bytes). Storage requirements depend on the number of ASCII characters in the program.

70000 — Memory allocation for Compiled Profiles & Programs (bytes). Storage requirements depend on the number of segments (1 segment consumes 72 bytes). A segment could be one of these commands:

Contouring:	Compiled Motion:	PLC (PLCP) Program:
PARCM	GOBUF *	IF **
PARCOM	PLOOP	ELSE
PARCOP	GOWHEN	NIF
PARCP	TRGFN	L
PLIN	POUTA	LN
	POUTB	OUT
	POUTC	ANO
	POUTD	EXE
	POUTE	PEXE
	POUTF	VARI **
	POUTG	VARB **
	POUTH	

* GOBUF commands may require up to 4 segments.
** IF statements require at least 2 segments; each AND or OR compound requires an additional segment. VARI and VARB each require 2 segments.

The table below identifies memory allocation defaults and limits for all 6K Series products. When specifying the memory allocation, use only even numbers. The minimum storage capacity for one partition area (program or compiled) is 1,000 bytes.

Feature	All Other Products
Total memory (bytes)	150,000
Default allocation (program,compiled)	149000,1000
Maximum allocation for programs	149000,1000
Maximum allocation for compiled profiles & PLC programs	1000,149000
Max. # of programs	400
Max. # of labels	600
Max. # of compiled profiles & PLC programs	300
Max. # of compiled profile segments	2069
Max. # of numeric variables (VAR)	225
Max. # of integer variables (VARI)	225
Max. # of binary variables (VARB)	125
Max. # of string variables (VARS)	25

When teaching variable data to a data program (DATP), be aware that the memory required for each data statement of four data points (43 bytes) is taken from the memory allocation for program storage (see *Variable Arrays* in Chapter 3, page 94, for details).

CAUTION

Using a memory allocation command (e.g., MEMORY80000,70000) will erase all existing programs and compiled profile segments & PLC programs. However, issuing the MEMORY command without parameters (i.e., type MEMORY *<cr>* to request the status of how the memory is allocated) will not affect existing programs or compiled segments/programs.

Checking Memory Status

To find out what programs reside in your controller's memory, and how much of the available memory is allocated for programs and compiled profile segments, issue the TDIR command (see example response below). Entering the TMEM command or the MEMORY command (without parameters) will also report the available memory for programs and compiled profile segments.

Sample response to TDIR command

```
*1 - SETUP USES 345 BYTES
*2 - PIKPRT USES 333 BYTES
*79322 OF 80000 BYTES (98%) PROGRAM MEMORY REMAINING
*972 OF 972 SEGMENTS (100%) COMPILED MEMORY REMAINING
```

Two system status bits (reported with the TSS and SS commands) are available to check when compiled profile segment storage is 75% full or 100% full. System status bit #29 is set when segment storage reaches 75% of capacity; bit #30 indicates when segment storage is 100% full.

Executing Programs (options)

Following is a list of the primary options for executing programs stored in your controller:

Method	Description	See Also
Execute from a terminal emulator	Type in the name of the program and press enter; or write a program to prompt the operator to select a program from the terminal.	-------
Execute as a subroutine from a "main" program	Use a branch (GOTO, GOSUB, or JUMP) from the main program to execute another stored program.	Page 22
Execute automatically when the controller is powered up	Assign a specific program as a startup program with the STARTP command. When you RESET or cycle power to the controller, the startup program is automatically executed.	Page 12
Execute from a PLC program	Write a PLC program that executes a program (using EXE or PEXE) based on a specific condition (e.g., input state). Use the SCANP command to launch the PLC program in the PLC Scan Mode.	Page 104
Execute a specific program with BCD weighted inputs	Define programmable inputs to function as BCD select inputs, each with a BCD weight. A specific program (identified by its number) is executed based on the combination of active BCD inputs. Related commands: INSELP and INFNCi-B or LIMFNCi-B.	Page 82
Execute a specific program with a dedicated input	Define a programmable input to execute a specific program (by number). Related commands: INSELP and INFNCi-iP or LIMFNCi-P.	Page 88
"Call" from a high-level program	Using a programming language such as BASIC or C, write a program that enables the computer to monitor processes and orchestrate motion and I/O by executing stored programs (or individual commands) in the controller.	Page 118
Execute from an RP240 (remote operator interface)	Execute a stored program from the RUN menu in the RP240's standard menu system.	Page 111
Execute from your own custom Windows program	Use a programming language (e.g., Visual Basic, Visual C++, etc.) and the 6K Communications Server (provided on the Motion Planner CD) to create your own windows application to control the 6K product.	---------

Creating and Executing a Set-up Program

The intent of the Setup program is to place the 6K controller in a ready state for subsequent motion control. The setup program must be called from the "main" program for your application; or you can designate (with STARTP) the setup program as the program to be is automatically executed when the 6K product is powered up or when the RESET command is executed. The setup program typically contains elements such as feedback device configuration, tuning gain selections, programmable I/O definitions, scaling, homing configuration, variable initialization, etc. (more detail on these "basic" features is provided in Chapter 3, *Basic Operation Setup*).

The basic process of creating a setup program is:

1. Create a program to be used as the setup program.
2. Save the program and download it to the 6K product.
3. Execute the STARTP command to assign your new program as the "start-up" program (e.g., STARTP setup assigns the program called "setup" as the start-up program). The next time the controller is powered up or reset, the assigned STARTP program will be executed.

Or call the setup program from the main program for your application.

Syntax -- Command Value Substitutions

Many commands can substitute one or more of its command field values with one of these substitution items (demonstrated in the programming example below):

- VARB .... Uses the value of the binary variable to establish <u>all</u> the fields in the command.
- VAR .... Places current value of the numeric variable in the corresponding field of the command.
- READ .... Information is requested at the time the command is executed.
- DREAD......... Reads the RP240's numeric keypad into the corresponding field of the command.
- DREADF ...... Reads the RP240's function keypad into the corresponding field of the command.
- TW................ Places the current value set on the thumbwheels in the corresponding field of the command.
- DAT .... Places the current value of the data program (DATP) in the corresponding field of the command.

Programming Example: *(NOTE: The substitution item must be enclosed in parentheses.)*

```
VAR1=15              ; Set variable 1 to 15
A5,(VAR1),4,4        ; Set acceleration to 5,15,4,4 for axes 1-4, respectively
VARB1=b1101XX1       ; Set binary variable 1 to 1101XX1 (bits 5 & 6 not affected)
GO(VARB1)            ; Initiate motion on axes 1, 2 & 4 (value of binary
                     ; variable 1 makes it equivalent to the GO1101 command)
OUT(VARB1)           ; Turn on outputs 1, 2, 4, and 7
VARS1="Enter Velocity" ; Set string variable 1 to the message "Enter Velocity"
V2,(READ1)           ; Set the velocity to 2 on axis 1. Read in the velocity for
                     ; axis 2 , output variable string 1 as the prompting message
                     ; 1. Operator sees "ENTER VELOCITY" displayed on the screen.
                     ; 2. Operator enters velocity prefixed by !' (e.g., !'20).
HOMV2,1,(TW1)        ; Set the home velocity to 2 and 1 on axes 1 and 2, respectively.
                     ; Read in the home velocity for axis 3 from thumbwheel set 1
HOMV2,1,(DAT1)       ; Set the home velocity to 2 and 1 on axes 1 and 2, respectively.
                     ; Read home velocity for axis 3 from data program 1.
```

Rule of Thumb

Not all of the commands allow command field substitutions. In general, commands with a binary command field (<b> in the syntax) will accept the VARB substitution. Commands with a real or integer command field (<r> or <i> in the syntax) will accept VAR, READ, DREAD, DREADF, TW or DAT.

Programming Error Messages *(continued)*

Error Response	Possible Cause
INVALID DATA	Data for a command is out of range. Following (these conditions can cause an error during Following): • The parameter supplied with the command is valid. FFILT.....Error if: smooth number is not 0-4 FMCLEN...Error if: master steps > 999999999 or negative FMCP.......Error if: master steps > 999999999 or < -999999999 FOLMD.....Error if: master steps > 999999999 or negative FOLRD.....Error if: master steps > 999999999 or negative FOLRN.....Error if: slave steps > 999999999 or negative FSHFC.....Error if: number is not 0-3 FSHFD.....Error if: slave steps > 999999999 or < -999999999 GOWHEN...Error if: position > 999999999 or < -999999999 WAIT.......Error if: position > 999999999 or < -999999999 • Error if a GO command is given in the preset positioning mode (MCØ) and: FOLRN = zero FOLMD = zero, or too small (see Following chapter in the *6000 Series Programmer's Guide*)
INVALID FOLMAS SPECIFIED	Following: An illegal master was specified in FOLMAS. A slave may never use its own commanded position or feedback source as its master.
INVALID RATIO	Following: Error if the FOLRN:FOLRD ratio after scaling is > 127 when a GO is executed
LABEL ALREADY DEFINED	Defining a program or label with an existing program name or label name
MAXIMUM COMMAND LENGTH EXCEEDED	Command exceeds the maximum number of characters
MAXIMUM COUNTS PER SECOND EXCEEDED	Velocity value is greater than 1,600,000 counts/sec
MOTION IN PROGRESS	Attempting to execute a command not allowed during motion (see Restricted Commands During Motion section in the *6000 Series Programmer's Guide.*) Following: The FOLEN1 command was given while that slave was moving in a non-Following mode.
NEST LEVEL TOO DEEP	IFs, REPEATs, WHILEs, & GOSUBs nested greater than 16 levels
NO MOTION IN PROGRESS	Attempting to execute a command that requires motion, but motion is not in progress
NO PATH SEGMENTS DEFINED	Compiled Profile compilation error
NO PROGRAM BEING DEFINED	END command issued before a DEF command
NOT ALLOWED IF SFBØ	Changes to tuning commands (SGILIM, SGAF, SGAFN, SGI, SGIN, SGP, SGPN, SGV, SGVN, SGVF, and SGVFN) and SMPER are not allowed if SFBØ is selected
NOT ALLOWED IN PATH	Compiled Profile path compilation error
NOT DEFINING A PATH	Executing a compiled profile or contouring path command while not in a path
NOT VALID DURING FOLLOWING MOTION	A GO command was given while moving in the Following mode (FOLEN1) and while in the preset positioning mode (MCØ).
NOT VALID DURING RAMP	A GO command was given while moving in a Following ramp and while in the continuous positioning mode (MC1). Following status (FS) bit #3 will be set to 1. A FOLEN command was given during one of these conditions: • During a shift (FSHFC or FSHFD) • During a change in ratio (FOLRN/FOLRD) • During deceleration to a stop
PATH ALREADY MOVING	Compiled Profile path compilation error
PATH NOT COMPILED	Attempting to execute a individual axis profile or a multiple axis contouring path that has not been compiled

Example:

```
IF(2V<25)      ; If the programmed velocity on axis 2 is less than 25
               ; units/sec, then do the statements between the IF and NIF
VAR1=2V*2      ; Variable 1 = programmed velocity of axis 2 times 2
V,(VAR1)       ; Set the velocity on axis 2 to the value of variable 1
NIF            ; End the IF statement
```

VAR Numeric Variable Assignment

Type	Variable
Syntax	`<!>VAR<i><=r>`
Units	i = variable number, r = number or expression
Range	i = 1 - 100 (AT6n00), 1-200 (AT6n00-M) i = 1 - 150 (AT6n50, 610n, 615n, 620n, 625n, & 6270), or 1 - 150 (AT6n50, 610n, 615n, 620n, 625n, & 6270), r = ±999,999,999.99999999
Default	n/a
Response	VAR1: *VAR1=+0.0
See Also	VARB, VARCLR, VARS, WRVAR

Product	Rev
AT6n00	1.1
AT6n50	1.0
610n	4.0
615n	1.0
620n	1.2
625n	1.0
6270	1.0

Numeric variables can be used to store any real number value, with a range from -999,999,999.99999999 to +999,999,999.99999999. The information is assigned to the variable with the equal sign (e.g., VAR1=32.3).

Stand-alone 6000 products: All variables (numeric [VAR], binary [VARB], and string [VARS]) are automatically stored in battery-backed RAM.

Variables are also used in conjunction with mathematical (=, +, -, *, /, SQRT), trigonometric (ATAN, COS, PI, SIN, TAN), and bitwise operators(&, |, ^, ~). For example, VAR1=(3+4-7*4/4+3-2/1.5)*3.

Each variable expression must be less than 80 characters in length, including the VAR1= part of the expression.

Numeric data can also be read into a variable, through the use of the READ , DAT, or TW commands (e.g., VAR1=READ1).

All variables can be used within commands that require a real or integer value. For example, the A command requires real values for acceleration, therefore the command A(VAR1),1Ø,12,(VAR2) is legal. Indirect variable assignments are also legal; (e.g., VAR(VAR1)=5 or VAR(VAR2)=VAR(VAR4)).

<u>Rule of Thumb for command value substitutions</u>: If the command syntax shows that the command field requires a real number (denoted by <r>) or and integer value (denoted by <i>), you can use the VAR substitution.

Example:

```
VAR1=2*PI          ; Set Variable 1 to 2p
D(VAR2),,(VAR3)    ; Set the distance value on axis 1 equal to variable 2,
                   ; and the distance on axis 3 equal to variable 3
```

Indirect Variables: Numeric variables can be used indirectly. Only one level of indirection is possible (e.g., VAR(VAR(VARn)) is not a legal command). The example below shows how indirect variables are used to clear 50 variables (from 1 to 50).

Example:

```
VAR51 = 1             ; Set Variable 51 to 1
REPEAT                ; Begin repeat/until loop
VAR(VAR51) = 0        ; Clear variables (e.g., if VAR51 = 8,
                      ; then VAR(VAR51)=0 is equivalent to VAR8=0)
VAR51 = VAR51 + 1     ; Increment counter
UNTIL (VAR51 = 51)    ; End repeat/until loop
```

VARB — Binary Variable Assignment

Type	Variable
Syntax	<!>VARB<i><=bb...bbb> (32 bits)
Units	i = variable number
Range	i = 1-100 (AT6n00); 1-25 (AT6n50, 610n, 615n, 620n, 625n, & 6270) b = 0, 1, X, or x
Default	n/a
Response	VARB1: *VARB1=XXXX_XXXX_XXXX_XXXX_XXXX_XXXX_XXXX_XXXX
See Also	VAR, VARCLR, VARS, VCVT, WRVARB

Product	Rev
AT6n00	1.0
AT6n50	1.0
610n	4.0
615n	1.0
620n	1.0
625n	1.0
6270	1.0

Binary variables can be used to store any 32-bit or less binary value. The 32-bit binary value must be in the form of 32 ones, zeros, or Xs. The information is assigned to the binary variable with the equal sign.

Stand-alone 6000 products: All variables (numeric [VAR], binary [VARB], and string [VARS]) are automatically stored in battery-backed RAM.

Example: `VARB1=b1111ØØØØ1111XXXX1111ØØØØxxxx1111`
Notice that the letter b is required. The b signifies binary, 1's, Ø's, and X's only.

Example: `VARB1=h7F4356A3`
Notice that the letter h is required. The h signifies hexadecimal, Ø-9, A-F only.

Binary variables are also used in conjunction with bitwise operators (&, |, ^, and ~).
Example: `VARB1=VARB2 | VARB3 & b1111ØØØØ11ØØ1`

The expression must be less than 80 characters in length, including the (VARB1=b or VARB1=h) part of the expression.

All binary variables can be used to set bits for commands that require at least 4 bits of binary information. For example, the OUT command requires 24 bits of binary information, therefore the command OUT(VARB1) is legal.

Rule of Thumb for command value substitutions: If the command syntax shows that the command field requires a binary value (denoted by <b>), you can use the VARB substitution.

Example:
```
VARB1=b1110 & hA    ; Binary variable 1 is set to binary 1110 bitwise
                    ; "AND"ed with hexadecimal A
VARB1=IN.7          ; Binary variable 1 is set to input bit 7
OUT(VARB1)          ; Assign the value of binary variable 1 to the outputs
```

VARCLR — Variable Clear

Type	Variable
Syntax	<!>VARCLR
Units	n/a
Range	n/a
Default	n/a
Response	n/a
See Also	VAR, VARB, VARS

Product	Rev
AT6n00	3.4
AT6n50	3.4
610n	4.0
615n	3.4
620n	3.4
625n	3.4
6270	3.4

VARCLR resets all numeric variables (VAR), binary variables (VARB), and string variables (VARS) to their factory default values:

Numeric variables are set to 0.0
Binary variables are set to bxxxx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx
String variables are set to ""

```
L5
WRITE "*BEG"
LN
VAR1=0           ; # # # # # # Start ROLLER ON
VAR2=1
VAR4={INCREMENT_AXIS_1}*{MULTIPLIER_AXIS_1} «1
VAR3={PROGRAM_LOOPS}
VAR5={FINAL_INCREMENT_AXIS_1}*{MULTIPLIER_AXIS_1} «1
OUT.3-1
IF({TRAVERSE_AXIS_3}>0)
OUT.6-1
WAIT(IN.5=B1)
OUT.6-0
VARB2=B1
ELSE
VARB2=B0
NIF
L(VAR3)
IF(VAR2>1)
D0
V5
G01X
Gosub Tinfo
NIF
VAR2=VAR2+1
```